EP 22373 B1

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/28, 17/27 — A1

(11) International Publication Number: **WO 00/11571**

(43) International Publication Date: 2 March 2000 (02.03.00)

(21) International Application Number: PCT/US99/19255

(22) International Filing Date: 20 August 1999 (20.08.99)

(30) Priority Data:

60/097,630	24 August 1998 (24.08.98)	US
60/105,428	23 October 1998 (23.10.98)	US
09/372,710	11 August 1999 (11.08.99)	US

(81) Designated States: CN, JP, RU, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

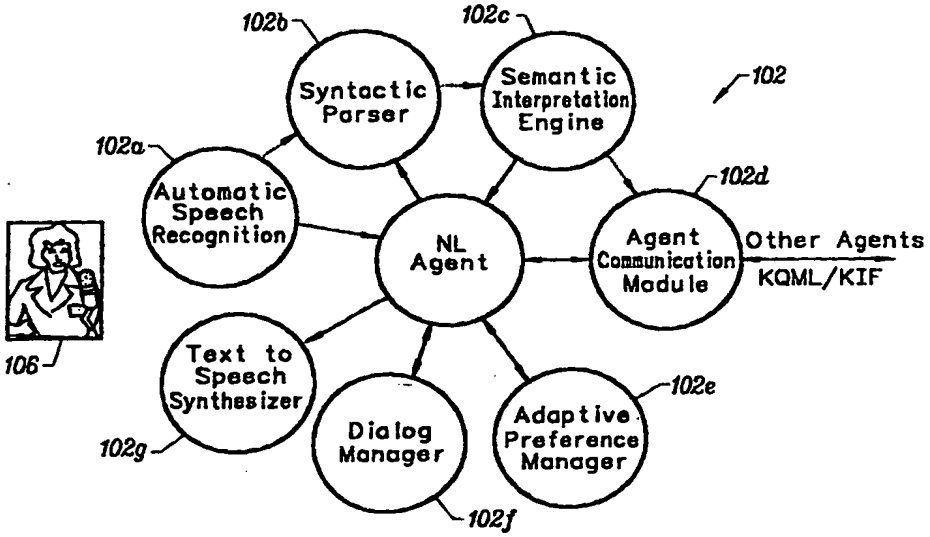
Published
With international search report.

(71) Applicant: BCL COMPUTERS, INC. [US/US]; 990 Linden Drive, Suite 203, San Jose, CA 95129 (US).

(72) Inventors: HARTONO, Rachmat; 3611 Madrid Drive, San Jose, CA 95132 (US). KHAN, Zeeshan; 1569 Tobias Drive, San Jose, CA 95118 (US). TJAHJADI, Timotius; 903 Sapphire Court, San Jose, CA 95136 (US). ALAM, Hassan; 1090 Leslie Drive, San Jose, CA 95117 (US). VU, Giac; 1449 East Calaverad Boulevard #2, Milpitas, CA 95117 (US).

(74) Agent: KUO, Jung-hua; Ritter, Van Pelt and Yi, Suite 205, 4906 El Camino Real, Los Altos, CA 94022 (US).

(54) Title: ADAPTIVE NATURAL LANGUAGE INTERFACE



(57) Abstract

A system and method for providing natural language interface for a computer system that interprets natural language user input and outputs responses using natural language are disclosed. The system (102) includes a natural language agent adapted to receive and interpret the natural language user input and to output an output command and at least one application agent adapted to receive and further interpret the output command from the natural language agent and to output an executable instruction to an application program. The natural language agent includes a syntactic parser (102b) adapted to generate a parsed sentence from the natural language user input, a semantic interpreter (102c) adapted to generate the output command from the parsed sentence, and an agent communication manager (102d) adapted to provide communication between the semantic interpreter. Each application agent may include a semantic task interpreter and at least one application wrapper.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

ADAPTIVE NATURAL LANGUAGE INTERFACE

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application claims priority to Provisional Patent Application Nos. 60/105,428 entitled "Adaptive Natural Language Interface for Use in Applications" filed on October 3, 1998, and 60/097,630 entitled "Adaptive Personal Assistant (APA)" filed on August 21, 1998.

STATEMENT OF RIGHTS TO INVENTIONS MADE UNDER FEDERALLY SPONSORED RESEARCH

The U.S. Government has a paid-up license in certain claims of this invention and the right in limited circumstances to require the patent owner to license others on reasonable terms as provided for by the terms of DARPA Contract Numbers DAAH01-96-C-R241 and DAAH01-99-C-R057.

BACKGROUND OF THE INVENTION

1. Field of Invention

The present invention relates generally to an adaptive natural language interface for use in applications. More specifically, the present invention provides a method for receiving commands, executing received commands and adaptively interacting with the user using a natural language interface, such as a natural language speech interface.

2. Description of Related Art

Making computers more user-friendly has long been a goal. More and more people, include people in non-technical fields and even young kids, use

computers for various purposes, such as personal, school and/or business. Computer systems are also handling more complex tasks resulting in more increasingly more complex operations. Even conceptually simple tasks require users to execute multiple complex steps for completion of the tasks.

Further, when a user switches between different application programs or vendors, e.g. from MICROSOFT EXCHANGE to NETSCAPE, the same conceptual task requires the operator to learn a new set of steps to complete the same task. For example, a conceptually simple task such as finding out whether the user has received a certain message, the user must be trained in the platform-specific graphical user interface of scrolling and the vendor-specific method of viewing new mail. As is evident, a conceptually simple task may require the user to execute multiple complex steps.

As the number of users and the complexity of computer systems increase, there is an increased need for computer systems and computer applications that require little or no training for the user to use. There is also an increased need for a method to efficiently and productively use, manipulate and control computers and applications running on computers.

Natural or spoken language is an efficient method for people to communicate and express commands. For example, voice-recognition method and software have been developed and are commercially available. Although some of these method and software allow the user to speak certain commands for the computer to execute, these voice-recognition method and software support only a predetermined set of commands at a very low-level of abstraction. The user must learn the precise words and syntax that the

software can accept. In other words, the voice communication cannot handle and interpret high-level, abstract, natural language commands.

Because natural language is an efficient and easy method for people to communicate and express commands, there is a long felt need for a voice-based command system and interface that can handle high-level, abstract commands and that responds to natural language.

The Air Force Institute of Technology, MIT Media Lab, Oregon Graduate Institute, Microsoft and IBM are examples of groups conducting research in the area of spoken language input. See, for example, Ball, "Mixing Scripted Interaction with Task-Oriented Language Processing in a Conversational Interface," International Conference on Intelligent User Interfaces, Jan. 5-8, 1999, Redondo Beach, CA, pg. 101-104.

U.S. Patent No. 5,748,974 assigned to IBM Corp. describes an example of spoken language input and, more specifically, a multimodal natural language interface for cross-application tasks. The multimodal natural language interface interprets user requests by combining natural language input from the user (spoken, typed or handwritten) with information selected from an application currently in use by the user to perform a task in another auxiliary application for processing. The information is selected by a standard technique from the current application.

Copending U.S. Patent Application Serial No. 08/919,138, assigned to the assignee of the present application and incorporated herein in its entirety by reference, describes a natural-language speech control method. The natural-language speech control method produces a command for controlling the operation of a computer from words spoken in a natural language. The

method includes processing an audio signal representing the spoken words of a user to generate textual digital computer data (e.g. ASCII text), processing the textual digital computer data with a natural language syntactic parser to produce a parsed sentence that includes a string of words with each word being associated with a part of speech in the parsed sentence, and generating the command from the parsed sentence.

SUMMARY OF THE INVENTION

The present invention comprises a method for receiving commands and/or adaptively outputting results and responses using a natural language interface, such as a natural language speech interface. The method utilizes an agent-based architecture comprising a front-end natural language agent and one or more application task agents for each class of applications.

It should be appreciated that the present invention can be implemented in numerous ways, including as a process, an apparatus, a system, a device, a method, or a computer readable medium such as a computer readable storage medium or a computer network wherein program instructions are sent over optical or electronic communication lines. Several inventive embodiments of the present invention are described below.

In one embodiment, the natural language interface for a computer system includes a natural language agent adapted to receive and interpret the natural language user input and to output an output command and at least one application agent adapted to receive and further interpret the output command from the natural language agent and to output an executable instruction to an application program. The natural language agent includes a syntactic parser

adapted to generate a parsed sentence from the natural language user input, a semantic interpreter adapted to generate the output command from the parsed sentence, and an agent communication manager adapted to provide communication between the semantic interpreter. Each application agent may include a semantic task interpreter adapted to generate the executable instruction from the output command of the natural language agent, and at least one application wrapper, each wrapper configured to communicate with a corresponding application program.

In another embodiment, a computer readable medium on which are stored natural language interface instructions executable on a computer processor is disclosed. The natural language interface instructions generally comprises receiving natural language user input, generating a parsed sentence from the natural language user input, mapping the parsed sentence into a semantic action, and generating an instruction from the semantic action, the instruction being executable by an application.

In yet another embodiment, a method for receiving, interpreting and executing natural language user input is disclosed. The method generally comprises receiving natural language user input, generating a parsed sentence from the natural language user input, semantically interpreting the parsed sentence and generating an output command from the parsed sentence, outputting the output command to an application class agent, semantically interpreting the output command and generating an executable instruction from the output command, and outputting the executable instruction to an application program for execution by the application program.

The present invention is a method for abstracting complex sequence computer operations into a conceptually simple task. The natural language interface parses the users' input and semantically maps it into a knowledge concept structure. The system then determines which application context should be responsible for interpreting and executing that command concept. The system utilizes task application wrappers to map the complex application tasks to vendor-specific executable tasks. Thus, the natural language interface system of the present invention allows users to control multiple desktop applications by abstract commands.

The system of the present invention lowers the barrier to entry to computing and greatly increases productivity by combining a spoken language system with the ability to handle higher order abstract commands in naturally spoken language. The system combines a spoken language interface with a knowledge-based semantic interpretation such that semantically equivalent abstractions result in the same operation. Syntactic and semantic interpretation of spoken language enable ease of use and complexity abstraction and provides the user access to computing through spoken language.

The system and method can be adapted to user preferences with feedback using active and passive relevance feedback techniques. Further, the present invention may include a natural language based help system in the natural language agent and each application class agent that collaborate with the user in offering assistance. For example, the system may prompt the user for semantically correct input, help the user complete tasks, and remind the user on tasks that need to be done.

The system of the present invention may be utilized and is compatible with existing software applications and platforms. The system uses a set of application class agents and wrappers that provide interface between the application class agent and different applications in the class. Each agent works with a class of applications, such as electronic mail, and communicates with specific applications through application wrappers. Thus, with a modular distributed agent architecture, the system and method of the present invention is extendable to multiple applications and is scalable to large sets of networked computer systems.

These and other features and advantages of the present invention will be presented in more detail in the following detailed description and the accompanying figures which illustrate by way of example the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a schematic illustration of the system and method of the present invention comprising an adaptive natural language interface for use in executing computer applications;

FIG. 2 is a schematic illustration of the natural language agent;

FIG. 3 shows a simplified model of a traditional dialog manager for ordering pizza through an interactive system;

FIG. 4 is a schematic illustration of the application class agent;

FIG. 5 illustrates the mapping of natural language into a set of semantic tasks by each task agent;

FIG. 6 illustrates an example of a personality assessment grid;

FIG. 7 illustrates an example of a computer system that can be utilized to execute the software of an embodiment of the invention and use hardware embodiments; and

FIG. 8 illustrates a system block diagram of the computer system of **FIG. 7**.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention comprises a system and method for receiving commands and/or adaptively outputting results using a natural language speech interface, such as a natural language speech interface. The system and method are an agent-based architecture comprising a front-end natural language agent and an application class task agents for each class of applications. The system and method may include adapting to each user, including the user's speech pattern, the current or recent commands issued by the user and the user's preferences. The following description is presented to enable any person skilled in the art to make and use the invention. Descriptions of specific embodiments and applications are provided only as examples and various modifications will be readily apparent to those skilled in the art. The general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the invention. Thus, the present invention is to be accorded the widest scope encompassing numerous alternatives, modifications and equivalents consistent with the principles and features disclosed herein. For purpose of clarity, details relating to technical material that is known in the technical fields

related to the invention have not been described in detail so as not to unnecessarily obscure the present invention.

Referring to the schematic illustration of **FIG. 1**, there is shown an adaptive natural or spoken language user interface system **100** for use in executing computer applications. The interface system **100** generally comprises a voice or front-end natural language agent **102** and one or more task agents **104a-d**. As is generally shown, the user **106** communicates an input phrase, command or sentence **108** to the natural language agent **102** which processes the input sentence and sends the input sentence to the appropriate one of the back-end application class task agents **104a-d**. Examples of the task agents **104a-d** shown in **FIG. 1** are meeting agent **104a**, personal information manager agent **104b**, email agent **104c**, and voice training agent **104d**. Each task agent **104a-d** outputs to the natural language agent **102** which then delivers the natural language output **110** to the user **106**.

Each of the back-end application class task agents **104a-d** works with a class of one or more existing computer applications. The interface system can be adapted to existing computer applications so that users can operate a computer using spoken language as well as other input devices such as keyboard and pointing devices, giving full multi-modal interface to existing computer applications.

Although the natural language user interface system **100** is generally described as one interacting in spoken natural language, the system **100** may be configured to receive and/or output using one or more alternative input and/or output mechanisms while utilizing natural language for such input and/or output interactions. Suitable alternative modes of input and/or output

include keyboard, mouse, touch or contact sensitive screen, and/or screen display.

FIG. 2 is a schematic illustration of the natural language agent **102**. The natural language agent **102** communicates with the user **106** through spoken language. The natural language agent **102** preferably includes:

- automatic speech recognition system **102a**;
- natural language syntactic parser **102b**;
- natural language semantic interpreter **102c**;
- agent communication manager **102d**;
- adaptive preference manager **102e**;
- dialog manager **102f**; and
- text-to-speech synthesizer **102g**.

The natural language agent **102** executes a first level interpretation of the natural language input. The front end natural language agent **102** receives all natural language input and determines which of the available task agents **104** to pass the natural language input interpreted by the front end natural language agent **102**. The task agent **104** to which the natural language input was passed may return a response such as an output to the front end natural language agent **102**. The front end natural language agent **102** then outputs the response from the particular task agent **104** to the user **106**. The natural language agent **102** may itself return a response if it determines that the original natural language input is incomplete, incorrect, or otherwise cannot be properly interpreted.

Each of these components **102a-g** of the natural language agent **102** are described in more detail below.

Automatic Speech Recognition System 102a

Automatic speech recognition systems for speech input are readily and commercially available off the shelf. Any suitable off the shelf speech recognition systems may be used as the automatic speech recognition system **102a** in the natural language interface system **100** of the present invention. Thus, details of speech recognition methods and systems are not described herein. In addition, error correcting techniques and cue words may be utilized to improve accuracy and allow for dialog management to effectively recognize speech input.

Natural Language Syntactic Parser 102b

There are generally three basic approaches to natural language syntactic processing: simple grammar, statistical and Government-and-Binding (GB-based). Simple grammar is used for simple, non-complicated syntax. Statistical approach examines word patterns and word co-occurrence and attempts to parse natural language sentences based on the likelihood of such patterns. Statistical approach uses a variety of methods such as neural networks and word distribution. The statistical approach is limited by an upper limit on error rate and it is very difficult to handle wide varieties of linguistic phenomena such as scrambling, NP (noun phrase) movement, binding between question words and empty categories.

The GB-based approach is described in, for example, "Some Concepts and Consequences of the Theory of Government and Binding," Cambridge, MA, MIT Press, the entirety of which is incorporated herein by reference. The GB-based approach is a more robust approach to natural language parsing using computational methods based on linguistic theory of a universal language. GB-based approach reveals implied syntactic structure in English language sentences and thus better facilitates in resolving ambiguous syntactic structures. By using generalized principles and parameters, the GB-based approach allows a customizable and portable parser that can be tailored to different environments and languages with little modification.

Preferably, the natural language syntactic parser **102b** utilizes a GB-based principle and parameters framework to parse natural language computer commands. Hageman, L. Introduction to Government and Binding Theory, incorporated by reference herein, for example, describes this concept. With the generalized principles and parameters, GB-based approaches can describe a large syntax and vocabulary relatively easily and thus may result in higher robustness than other approaches. With a GB-based approach, commands to computers can be seen as verb phrases that are a sub-set of complete English sentences. The sentences have an implied second person singular pronoun subject and the verb is active present tense.

For example, to resume work on a previous project, the user **106** may state "show me the first message." This request would be parsed into the following structure:

(VP (Vbar (V (V_IP

(V_IP show [present sg])

```
(IP
 (NP (Nbar (N me [goal animate sg])))
 (Ibar (NP [these inanimate sg]
  (Det the)
  (Nbar
   (AP (Abar (A first)))
   (N message)))))))))
```

The parse would allow the computer to map the verb into a computer command action, with the noun phrase (NP) as the object and the adjective phrase (AP) as properties of the object.

Natural Language Semantic Interpreter 102c

The natural language semantic interpreter or interpretation engine **102c** is preferably a frame-based command interpretation system. The natural language semantic interpreter **102c** may interpret the syntactic parse using context sensitive methodologies. The natural language semantic interpreter **102c** uses a knowledge base populated with concept-interfaces that each application can handle. The natural language semantic interpreter **102c** takes the syntactic parse of a spoken language request and maps it into a generic concept frame used to invoke the appropriate application method. **TABLE I** lists examples of concept-interfaces.

TABLE I

ACTION-CONCEPT	TOPIC CONCEPT	APPLICATION-CONTEXT
Show	Email	Email application
Show	Email address	Address book application
Delete	Email	Email application
Show	General help	Natural language agent

Requests input to the computer are preferably transformed by the semantic interpretation engine **102c** from a syntactic parse into a variable length verb-head frame. The process has variable length noun phrases as arguments. The noun phrases in turn have arguments that are adjective phrases. The verb-head describes an action-concept. The noun phrases describing the objects on which the actions are performed are topic-concepts and the adjective phrases describing the type of objects are modifier concepts.

Reverse Grammar Generation Mechanism

The semantic interpretation engine **102c** may also include a reverse grammar generation mechanism. The reverse grammar generation mechanism may be implemented in each agent, i.e. the natural language agent and/or each of the task agents. The reverse grammar generation mechanism includes a list or vector for each word and corresponding probabilities for each word in the list. For example, for the word "I," "eye," or "aye," the associated vector or list includes those words, i.e. "I," "eye," and "aye," and may have corresponding probabilities of 80%, 15%, and 5%. These probabilities may be predetermined and may be adjusted depending upon each user's selection of words used or depending upon a subset or all of the users' selection of words used.

Upon receiving the syntactic parse of the spoken language request, the semantic interpretation engine **102c** determines the permutations of the syntactic parse using the list for each word. For example, using the exemplary vector above and ignoring the lists for all the other words, if an input request is "I want to go home," the permutations of the syntactic parse may include:

"I want to go home,"

"Eye want to go home"; and

"Aye want to go home."

Using the permutations, the semantic interpretation engine **102c** determines which words best fit the grammar of the syntactic parse. To determine each best fit word, the word with the highest probability, i.e. "I" in the example above, is evaluated and determined if that word is suitable given the context. If the word is not suitable given the context, the next word having each next highest probability is then evaluated and determined if that word is suitable given the context, until a suitable word is determined. Of course, if no suitable word is determined, then the natural language agent may request clarification or correction from the user.

A combination of the action-concept and the topic-concepts are used to determine which task agent should handle the request. If the request is for a specialized task agent, the request is routed to that specialized task agent. If the request is for the natural language agent **102** itself, a routine associated with the command is invoked with the topics and modifiers as arguments. Using the arguments for routing commands allows for better disambiguation than with the verb alone.

The above-described interpretation approach has the advantage of allowing the natural language agent **102** to query the user for clarification, for example, if the original request is incomplete, or otherwise cannot be properly interpreted by the natural language agent **102**. For instance, if arguments do not match the verb, then a clarification can be requested by the natural language agent **102**.

Further, the above-described interpretation approach has the advantage of allowing the natural language agent **102** to properly interpret natural language requests without requiring the user to input the request conforming to specific structures. For example, in requesting an airline ticket from Portland to Boston, the user may state "I'd like a ticket to Boston from Portland" or "I'd like a ticket from Portland to Boston." In response, the natural language agent **102** may request clarification as to Portland, Oregon or Portland, Maine, for example. The above-described interpretation approach has the advantage of that it does not rely upon certain key words in order to properly interpret the user's requests. Further, the interpretation technique may be context base or context sensitive.

Agent Communication Manager 102d

The application class task agents **104** may communicate with each other, preferably using Knowledge Query Manipulation Language (KQML) or any other suitable language, via the agent communication manager or module **102d**. The contents of a message between application class agents **104** may be coded in any suitable format, preferably the Knowledge Interchange Format (KIF). When a KQML message with an "achieve" performative is received by an agent **104**, the KIF encoded concept structure is interpreted further by the agent **104** through a semantic interpretation knowledge base similar to the one described above with reference to the semantic interpreter **102c**. In this case, the knowledge-base only includes information on how to map application-specific modifiers to application-task parameters. Using KQML and KIF allows the different agents **104** to easily communicate with each other. In

particular, the natural language agent **102** sends the user's request to the application class agent **104** via the agent communication manager **102d** and the application class agent **104** sends requests back to the natural language agent **102,** or some other agent, via the agent communication manager **102d**. Thus an email class agent **104c** can request information from a file manager class agent (not shown) using a KQML/KIF statement via the agent communication manager **102d**.

The above-described approach has the advantage of modularizing the different ontologies by allowing different application class agents **104** to have different subset dictionaries and task specific semantic interpretation knowledge-bases. It also allows the class agents **104** to handle vendor-specific application features by easily modifying the local semantic interpretation tables. This is described in more detail below with reference to the application class agent **104**.

The task routing mechanism is similar to Galaxy II, a voice controlled system integrating three separate voice controlled systems, as discussed in Seneff et al. "Galaxy-II: A Reference Architecture for Conversational System Development," 5th International Conference On Spoken Language Processing, Nov. 30 – Dec. 4, 1998, Sydney, Australia, p.931, the entirety of which is incorporated by reference herein. Currently, the Galaxy II requires a user to explicitly switch from one domain to another.

Adaptive Preference Manager 102e

The adaptive preference manager **102e** is associated the natural language each agent **102** and with each user **106**. The task of the adaptive

preference manager **102e** is to learn what default conditions are preferred the user **106** by either monitoring the users action implicitly (i.e. observing in the background) and/or by being instructed explicitly by the user **106** on positive and/or negative preferences. These preferences may be shared by different users **106** running similar application class agents **104**.

The adaptive preference manager **102e** uses relevance feedback techniques. Relevance feedback technique is widely used for preference optimization with declarative preferences. A request for executing an action based on preferences can be modeled as a query to locate a document in a collection of documents. In this technique widely used in information retrieval, the relevance of a document to a query is measured by how many matches the document has with query terms. In the realm of preference requests, the result of an action is analogous to a document where the preference is analogous to a query. Using this substitution, information retrieval techniques for ranking results of action requests can be adapted according to user preferences. Criteria specified in the spoken request are also factored as preferences. For preference matching, the information retrieval formula can be adapted for preference ranking by simplifying the equation for small queries as expressed in equation (1):

$$\text{similarity (Q, D)} = \frac{\sum_{i=1}^{t}(IDF_i * w_{ij})}{(\sum_{i=1}^{t}(IDF_i)^2 * \sum_{i=1}^{t}(w_{ij})^2)^{1/2}} \qquad (1)$$

where

t = total number of independent terms

w_{iq} = $(.5 + (.5\ qfreq_{iq}/maxfreq_q))$ x IDF_i

w_{ij} = $dfreq_{ij}$ x IDF_i

$qfreq_{iq}$ = Frequency of term i in request q

$dfreq_{ij}$ = Frequency of term i in result j

$maxfreq_j$ = Maximum frequency of any term in query

$maxfreq_q$

IDF_i = $\log_2 (maxn/n_i) + 1$

N = Number of results

n_i = Total number of occurrences of term i in the results

maxn = Maximum frequency of any term in the results

The qualitative ranking can be quantified by adding a set of weights to the ranking equations (2) and (3) as set forth below to incorporate the weights applied to terms in the definition of IDF_i.

$$max_n = \max ((\sum_{i=1}^{t} (w_i * n_j)) \forall j \quad (2)$$

$$n_i = \sum_{j=1}^{t} (w_j * n_i) \quad (3)$$

Relevance feedback techniques have been used in information retrieval techniques for improving the precision and recall of queries. In relevance feedback the query terms are reweighted by the selection of the retrieved items by the user. For the case where the user does not exhaustively select all the relevant responses, the reweighting of the term weights can be done by equations (4) and (5).

Initial Weights $W_{ijk} = (C + IDF_i) * f_{ik}$ (4)

Feedback $W_{ijk} = (C + \log p_{ij} (1-q_{ij})/(1-p_{ij})q_{ij}) f_{ik}$ (5)

where:

W_{ijk} = weight for term i in preference j and result k

IDF_i = the IDF weight for the term i in the entire set of result

p_{ij} = probability of the term i within the set of relevant results for preference j

q_{ij} = probability that term i is assigned with the set of non relevant results for preference j

$f_{ik} = K + (1 - K)* freq_{ik}/maxfreq_k$

$freq_{ik}$ = the frequency of term i in result k

$maxfreq_k$ = maximum frequency of any term in result k.

As noted above, the execution of a task with variable parameters can be modeled as an information retrieval query. In this case the weights for the query term can be modeled as the user's preference weights.

Help System

With a natural language based system that abstracts the semantic concepts from the complexity of tasks, much of the help system is implicitly encoded in the knowledge base. Instead of asking "How can I send my spreadsheet to John", the user asks the natural language agent **102** to "Send the spreadsheet to John." If invalid parameters are given, the user **106** is prompted for the correct parameters. However, the natural language interface system **100** is also able to handle requests for help by generating an explanation of how the request functions. It can also show an example of a typical user request to accomplish the task.

Dialog Manager 102f

The natural language agent **102** further includes a dialog manager **102f**. The dialog manager **102f** of the natural language agent **102** controls the interactions between the user **106** and the natural language interface system **100**. The dialog manager **102f** is an finite state machine (FSM) similar to the one described in Cohen, et al. "The Efficiency of Multimodal Interaction: A Case Study," 5th International Conference On Spoken Language Processing, Nov. 30 – Dec. 4, 1998, Sydney, Australia, p.253, incorporated herein in its entirety by reference.

The dialog manager **102f** handles tasks such as accepting user inputs, obtaining parameter for tasks, requesting clarification and asking for confirmation on tasks.

The ability to handle natural language commands extends the concept of traditional dialog managers. Traditional dialog managers function like finite state machines (FSM) accepting dialog. For example, as shown in **FIG. 3**, ordering a pizza through an interactive system requires the user to specify the type of pizza, such as the size and topping of the pizza. A simplified model may be adopted where the user must select the size of the pizza (small, medium, or large) and the toppings (cheese, Hawaiian or pepperoni), and confirm the order. If changes are to be made to the size while selecting the topping, then either this ability to make such a change must to be written into the FSM or the user must to wait until the end of the ordering sequence.

In contrast, with spoken language commands, many of these dialog steps are unnecessary. Some FSMs can be generalized to a set of Boolean operations on a set of choices. In this case choosing a pizza is an AND

operation (size, topping and confirmation) on a set of XOR operations (e.g. small, medium or large size). Thus in spoken natural language, the user may simply say "I wish to order a large cheese pizza."

As is evident, one natural language sentence completes all the choices and only a confirmation is necessary. However, additional dialog issues are created under various circumstances. For instance, the user may make an incomplete request such as "I want a cheese pizza," make an incorrect request such as "can you send veggie pizza," request information such as "what types of pizza do you have," change a request "I'd like to make that a small one," or make an out of context request such as "I want to see my email."

A global state variable may be introduced to allow the dialog manager **102f** the flexibility to handle such spoken language requests. The global state variable uniquely identifies the state of the interaction between the user **106** and the natural language agent **102**. The state of the natural language agent **102** can be in one of two classes: IDLE or DEFINED. If the natural language agent **102** is in IDLE, the natural language agent **102** is not actively engaged in a dialog with the user **106** and interprets the request in the default global context. If the natural language agent **102** is in a DEFINED state S1, the designer of the dialog has the option of specifying a set of semantic frames it will accept and the actions. If the semantic frame is not defined, the action would be deemed out of context.

With the above-described scheme, if an incomplete request was made, the user **106** is prompted for more information; if an incorrect request is made, the user **106** is given a set of options from which to chose; and if a change request is made, the order is changed; if an out of context request is made,

then the user **106** is asked if a context switch is indeed desired with a warning that the current context will be lost.

Text-To-Speech Synthesizer 102g

The natural language agent **102** may offer the user **106** the option of receiving messages as text-on-screen or as synthesized speech with text-to-speech synthesizer **102g**. The text-to-speech synthesizer **102g** preferably uses commercial off the shelf technology to communicate messages to the user **106** by speech. The text-to-speech synthesizer **102g** may utilize intonation to make the synthesized speech sound more natural to the user **106**. In addition, the natural language interface system **100** may use Avatars for output. The text and speech messages are transmitted in conjunction with other graphical items that may be displayed by the applications and/or the agents.

Application Class Agent 104

As shown in **FIG. 4** and described above, the agent communication module **102d** of the natural language agent **102** allows communication between the application class agents **104** and the natural language agent **102**. Each application class agent **104** preferably works with a single class of applications **112** that have similar conceptual operations. For example, different email applications generally perform the same conceptual actions of sending and receiving mail but performs these actions through different sets of steps.

Each application class agent **104** preferably includes a set of application wrappers **104A**, a semantic or task interpretation engine **104B**, an

application class communication or dialog manager **104C**, an adaptive application class preference manager **104D**, and an application class help system (not shown).

The communication between the application class agent **104** and each different type of vendor-specific applications programs **112** is via an application wrapper **104A** that translates the conceptual action to a set of application specific operations. The task application wrapper **104A** is the interface between the application class agent **104** and different applications **112** in the class. With wrappers **104A**, the application class agent **104** communicates with specific applications **112** allowing the incorporation of existing applications into the architecture of the system **100**. For example, an e-mail agent would have a wrapper for interacting with each email system such as NETSCAPE and MICROSOFT EXCHANGE.

To interface with existing applications, the wrapper **104A** is preferably written in one of the platform specific macro languages. Examples of platform specific macro languages are listed in **TABLE II**.

TABLE II

PLATFORM	MACRO LANGUAGE
MICROSOFT WINDOWS/95/98/NT	VISUAL TEST
MICROSOFT COM Compliant applications	MICROSOFT COM
X WINDOWS Applications	XTCL, XTK, PERL
Applications with API	API calls

The task or semantic interpretation engine **104B** is similar to the semantic interpretation engine **102c** of the natural language agent **102** described above. The task interpretation engine **104B** serves as the knowledge base for each agent **104**. The task interpretation engine **104B** receives the semantic frame representation as input. Based on the frame's head verb

(action request) and noun phrases (parameters), the task interpretation engine **104B** invokes a routine that sends a set of requests to the task application wrapper **104A**.

The application class dialog manager **104C** is similar to the natural language agent dialog manager **102f** of the natural language agent **102** described above. The application class dialog manager **104C** manages the interaction between the user **106** and the application class agent **104**, requests clarification for ambiguous requests, asks for confirmation, and obtains incomplete parameters.

The application class adaptive preference manager **104D** records the user-preferences for each task. The preference is computed in a manner similar to the general natural language agent preference calculation for the natural language agent adaptive preference manager **102e** as described above.

While the natural language capability of the natural language interface system **100** desirably removes most of the user's need for help, each application class preferably has a help capability to enhance the minimum training feature of the natural language interface system **100** of the present invention. The help system can be encoded in the application class interpretation engine **104B** such that the request will result in communications of instructions and explanation from the application class agent **104**. For example, requests such as "How do I," "Can you show me," "What are the possible values for" will result in a response from the application class agent **102** with instructions and explanation on how to perform the task.

The help system may provide various types of help information. The help system may provide description of the agent capabilities such as the

general uses of the application and the tasks the agent can perform. While the natural language interface system **100** is designed for unconstrained input, ambiguity resolution may require constraints in syntax and the help system may provide syntax for different tasks to the user **106**. Thus, if the user **106** is unable to get the application class agent **104** to perform a task, the user **106** may ask how to execute an operation. The help system can respond with a sample natural language sentence. In addition, the help system can also provide suitable parameter values and ranges as well as the typical general help information normally included with the application on, for example, how to use the specific application

Example: Address Book Agent

The operation of the system **100** will be brief described with reference to an address book agent as an example. The address book agent comprises a task interpretation engine, a dialog manager and one or more task wrappers. The typical key actions of an address book include show (to display all or part of an address), change (to change all or part of an address), add (to add a new address), delete (to delete an existing address), sort (to arrange address by a given category), open/close (to open or close an address book), save (to save an address book), copy/paste (to copy and paste data from one part of an address book to another part).

These actions can be interpreted by the address book agent with reference to a semantic frames knowledge-base. The frames are also inserted in the natural language agent routing table. An example of a listing of such frames is shown in **TABLE III**. An application wrapper interfaces with the

particular address book application. The routines will handle the tasks as described above and will interface to the address book module for, e.g., MICROSOFT EXCHANGE, and NETSCAPE.

Semantic Mapping

FIG. 5 schematically illustrates the mapping of the user's input phrase, command or sentence within a large set of syntactically correct natural language phrases, commands or sentences **140** into a set of semantic tasks or actions **142** by the semantic mapper **144**. Preferably, a semantic mapper **144** is provided for the natural language semantic interpreter **102c** of the natural language agent **102** and/or the semantic interpretation engine **104B** of each application class agent. For example, a different semantic mapper **144** may be provided for word processing applications, e-mail applications and spreadsheet applications. **TABLE IV** provides an illustrative listing of task agents for a class of applications and a list of sample tasks corresponding to each task agent.

TABLE III

Action-Concept (Verb)	Topic Concept (Main Noun Phrase)	Second Noun Phrase	Application-Context (State)	Routine
SHOW	EMAIL	JOHN	EMAIL-APPLICATION	Pointer to routine for showing John's email
SHOW	EMAIL-ADDRESS	Current	ADDRESS BOOK-APPLICATION	Pointer to routine for showing current email
DELETE	EMAIL	LAST	EMAIL-APPLICATION	Pointer to routine to delete last email

Each task agent for a class of applications is preferably provided with its own set of semantically correct sentences, semantic actions and semantic

mapping. Each task agent thus serves as the common user interface for the corresponding class of applications under the assumption that each application within the class accomplishes the same or generally overlapping set of tasks. In other words, in a given class of applications, there is a finite and relatively small set of semantically equivalent actions or tasks **142** which may be performed by each application within the class.

TABLE IV

TASK AGENT	SAMPLE TASKS
Mail	Sends, receives, composes and views email
Fax	Sends, receives, composes and views faxes
Letter	Composes, writes, and send letters
File	Manages files
OS	Manages OS, configuration, performance
Address	Manages Address books
Games	Plays and operates specific games
Flight Simulators	Runs flight simulators
Vehicle Simulators	Operates Land Vehicle Simulators (cars, bikes, tanks, etc.)
Naval Simulators	Operates water-based simulators (ships, subs etc.)
Sports Simulators	Operates Baseball, Soccer, Football etc. simulators
War Game & Strategy Simulators	Operates turn and real time-based war games, single and multi-player
Role Play Simulators	Operates avatar-based role playing games such as DOOM, TOMB RAIDER, ADVENTURE, ZORK
Action Simulators	Operates action games
PIM (personal information manager)	General interface to task, calendar, address book, and notebook manager
Printer	Selects and configures printers, prints documents
Calendar	Manages calendar, sets meetings and ticklers
Terminal	Connects to remote systems, logs on, logs off
Travel	Arranges travel
Encyclopedia	Searches for and displays information from encyclopedic literature
Image Viewer	Views and manipulates images
C++	Assists in managing and writing C/C++ Programs
Basic	Assists in managing and writing Basic Programs
GUI	Manipulates, configures and arranges Graphical User Interface
Presentation	Draws, arranges and manipulates slide presentations
Charting	Charts sets of numbers in various graphs and charts

TABLE IV	
TASK AGENT	**SAMPLE TASKS**
Meeting	Arranges and schedules meetings
Scheduler	Schedules tasks on the computer
Telephone	Dials out and receives calls; integrates with address book
Voice Mail	Sends, receives, plays, manages voicemail
Word Processor	Writes, prints, manipulates, formats documents
Spreadsheet	Writes, prints, manipulates, formats numerical data
Drawing	Draws, manipulates, formats diagrams, merges predrawn images
Web	Connects, navigates, searches the internet/world-wide web
Network	Connects networks, manages connections
Mathematical	Manages mathematical and scientific manipulation of numeric and formulaic data
Directory Assistance	Locates telephone numbers and addresses over the internet
Internet Retail Sales	Describes objects and sells to customers over the internet
Common Household Utility Agent (e.g., VCR, Toaster, HVAC)	Controls household devices
K-12 Education on, e.g. Physics, Chemistry, Math	Teaches, runs games, quizzes, etc. for mathematics and science subjects
General Education on History, Economics, Philosophy	Teaches specific liberal arts and humanities courses
Hands-on training for job-based tasks	Trains users to operate equipment
Internet Event lookup	Locates events, such as conferences, meetings, concerts and festivals, through the internet
Internet Product-Information lookup	Finds products and prices through the internet
Internet-based Meeting scheduler	Schedules meetings over the internet
Hardware manager	Manager Computer Hardware (screen, disk, etc.)

For example, for the word processor class of applications, a user may input "compose a letter to John Smith," "please begin drafting a letter to John Smith," or "can you create a letter for my friend, John Smith?", each of which is a syntactically correct sentence within the large set of syntactically correct sentences **140**. These user commands are semantically equivalent. In each

case, the semantic mapper **144** maps the user input to a specific action within the small set of semantic actions **142**. In this example, the semantic mapper **144** maps each of these user inputs to the same action, draft a letter to John Smith, and the same task is performed. Thus, the semantic mapper **144** ensures that the same task is performed in a given class of applications regardless of the specific user input.

Each application in the class may have a different method for accomplishing the same semantic task. In response to any of the user inputs in the example above, a word processor application composes or drafts a letter to John Smith although the particular word processor application may utilize an approach different from the approach used by another word processor application. By using a core set of semantically equivalent tasks **142** for each class of applications, the present invention allows the user to accomplish the same semantic task independent of the specific application utilized.

Although a single task agent is preferably provided for each class of applications, the task engine of each task agent includes a process-specific execution module for each application. For example, the word processing task agent may include an execution module for MICROSOFT WORD and another execution module for WORD PERFECT. The process-specific execution module translates the semantic action for the specific application.

The semantic mapper **144** is capable of reducing idiomatic sentences and output a mapped semantic action. Input sentences may be generally classified as wh-query, request, let construction, infinitive, embedded clause, semantic mappings and context-dependent. Examples of the classifications of input sentences are shown in **Table V**. Regardless of the classification of the

input sentence, each input sentence is mapped into a semantic action. Preferably, each mapped semantic action is in the form of a verb phrase or the imperative case with an implied non-phrase. "Show me mail message" is an example of a imperative verb phrase having "you" as the implied non-phrase.

TABLE V

Wh-query	
What I wouldn't give to see the back of a blue car	
What is stopping you from showing me my mail	
What would I give to see my mail	
Why don't you show me my mail	How is my email situation today
Why can't I see my mail	Why don't you clean my mailbox
Where is my mail	How about attaching this file
Request	
Can I see my mail	May I see may mail
Can you show me my mail	Would you allow me to see may mail
Can you like show me my mail	Will you show me my mail
Let construction	
Let me see my mail	Let me mail be seen by me
Let me mail be shown	Let me know when I get mail
Infinitive	
I want to check out my mail	I want to know if I got new mail
I would love to check out my mail	I want to access my mail
We want to see our mail now	He wants to see his mail
I need to see my mail now	Susan wants to see her mail
Embedded clause	
I will be upset if you do not show me mail right now	
I would really appreciate it if you could show me my mail	
I think it would be great if you could show me my mail	
I want you to tell me if I have new mail	
I'm looking to see whether I have new mail	
we would like it if you could show us our mail	
I want you to inform me if I have new mail	
Semantic mappings	
I would like for you to give me a view where I can see my mail	
I would love it if I could play with my mail	
Can you show me something like mail	
Can you show me what evertone has sent to me	
Context-dependent	
Let me see what you have	I want you to summarize these
What else can you do	Who sent the last one
I want to know all about the things you can do	

In addition to the various input sentences, the spoken input sentences **108** given by the user **106** may contain one or more of several types of errors which may occur. These errors include unrecognized word, bad parse, unhandled verb, unhandled object, unhandled verb/object attribute and/or task-specific error. Some errors may be better handled and addressed at the natural language agent **102** while other errors may be better handled and addressed at the appropriate task agent **104**. For example, errors relating to unrecognized word, bad parse and unhandled verb are preferably handled and addressed at the natural language agent **102**. Errors relating to unhandled object may be handled and addressed at either the natural language agent **102** or the task agent **104**. Further, errors relating to unholdled verb/object attribute and task-specific errors are preferably handled and addressed at the task agent **104**.

As discussed above, the interface **100** of the present invention is an adaptive natural language interface **100**. The output of the natural language agent **102** is preferably adaptive to the personality of the user **106** by first identifying the personality type, personality trait or characteristics of the user and utilizing that identification for responding to the user. **FIG. 6** illustrates an example of a personality assessment grid where users may be groups into one of four types: analytical, driver, amiable and expressive, which are defined depending upon the relative levels of assertiveness and responsiveness. The natural language agent may make a determination as to which of the four types best characterizes the user from factors such as the user's tone, pitch, speed and the actual words used by the user. Of course, the natural language agent may utilize any other factors, personality assessment methods and/or personality characterization schemes.

The natural language agent **102** is adaptive in that it utilizes that determination of the user **106** in responding to the user by delivering the output to the user or in requesting additional information from the user using simplified emotional response. The determination thus may affect the tone, pitch, speed and/or the actual words used to respond to the user. For example, in delivering the output to the user or in requesting additional information from the user, the natural language agent may be empathetic, for example, and express similar levels of assertiveness and/or responsiveness by varying the words used, the speed at which the words are delivered, the tone and/or the pitch of the words. In addition, the form of as well as the specific graphical interface seen by the user may be determined by the user, the application currently utilized and/or based upon the determination of the user's personality.

Although the adaptive natural or spoken language user interface system **100** is described above in terms of natural language speech input, the interface system can also recognize and interpret natural language non-speech command, such as text. The natural language interface is preferably embodied in a computer program product in the form of computer coded instructions executable by a computer processor and stored in a computer readable medium.

FIG. 7 illustrates an example of a computer system that can be utilized to execute the software of an embodiment of the invention and use hardware embodiments. **FIG. 7** shows a computer system 201 that includes a display 203, screen 205, cabinet 207, keyboard 209, and mouse 211. Mouse 211 can have one or more buttons for interacting with a GUI. Cabinet 207 houses a

CD-ROM drive and/or a floppy disc drive 213, system memory and a hard drive (see **FIG. 8**) which can be utilized to store and retrieve software programs incorporating computer code that implements aspects of the invention, data for use with the invention, and the like. Although a CD-ROM and a floppy disk 215 are shown as an exemplary computer readable storage medium, other computer readable storage media including magnetic tape, flash memory, system memory, RAM, other types of ROM, and hard drive can be utilized. Additionally, a data signal embodied in a carrier wave (e.g., in a network including the Internet) can be the computer readable storage medium.

FIG. 8 shows a system block diagram of computer system 201 used to execute a software of an embodiment of the invention or use hardware embodiments. As in **FIG. 7**, computer system 201 includes monitor 203 and keyboard 209, and mouse 211. Computer system 201 further includes subsystems such as a central processor 251, system memory 253, fixed storage 255 (e.g., hard drive), removable storage 257 (e.g., CD-ROM drive), display adapter 259, sound card 261, transducers 263 (speakers, microphones, and the like), and network interface 265. Other computer systems suitable for use with the invention can include additional or fewer subsystems. For example, another computer system could include more than one processor 251 (i.e., a multi-processor system) or a cache memory.

The system bus architecture of computer system 201 is represented by arrows 267. However, these arrows are illustrative of any interconnection scheme serving to link the subsystems. For example, a local bus could be utilized to connect the central processor to the system memory and display adapter. Computer system 201 shown in **FIG. 8** is but an example of a

computer system suitable for use with the invention. Other computer architectures having different configurations of subsystems can also be utilized.

While the preferred embodiments of the present invention are described and illustrated herein, it will be appreciated that they are merely illustrative and that modifications can be made to these embodiments without departing from the spirit and scope of the invention. Thus, the invention is intended to be defined only in terms of the following claims.

CLAIMS

What is claimed is:

1. A natural language interface for a computer system that interprets natural language user input, the natural language interface comprising:

a natural language agent adapted to receive and interpret the natural language user input and adapted to output an output command; and

at least one application agent adapted to receive and further interpret the output command from the natural language agent and adapted to output an executable instruction to an application program,

the natural language agent including:

a syntactic parser adapted to generate a parsed sentence from the natural language user input,

a semantic interpreter adapted to generate the output command from the parsed sentence, and

an agent communication manager adapted to provide communication between the semantic interpreter and the at least one application agent,

each of the at least one application agent including:

a semantic task interpreter adapted to generate the executable instruction from the output command of the natural language agent, and

at least one application wrapper, each wrapper configured to communicate with a corresponding application program.

2. The natural language interface of claim 1, wherein the semantic interpreter of the natural language agent includes a semantic mapper adapted to map the parsed sentence to a semantic action as the output command.

3. The natural language interface of claim 1, wherein the natural language agent further includes a speech recognition system adapted to receive and recognize natural language user speech input and to generate the natural language user request therefrom.

4. The natural language interface of claim 1, wherein the natural language agent further includes a dialog manager adapted to provide feedback to the user indicating the natural language agent's understanding of the natural language user input and to interact with the user in natural language to clarify the natural language user input as necessary.

5. The natural language interface of claim 4, wherein the natural language agent further includes a text to speech synthesizer adapted to provide the speech feedback to the user in speech.

6. The natural language interface of claim 1, wherein the natural language agent further includes an adaptive preference manager adapted to generate default conditions preferred by the user, the default conditions being specific to each user and/or common to multiple users.

7. The natural language interface of claim 1, wherein the semantic task interpreter of each application agent further includes a semantic mapper for mapping the output command to a semantic action as the executable instruction.

8. The natural language interface of claim 1, wherein each of the at least one application agent further includes a dialog manager adapted to provide natural language feedback to the user indicating the application agent's understanding of the natural language user input and to interact with the user in natural language to clarify the natural language user input as necessary.

9. The natural language interface of claim 1, wherein each of the at least one application agent further includes an adaptive preference manager adapted to generate default conditions preferred by the user for the specific application, the default conditions being specific to each user and/or common to multiple users.

10. The natural language interface of claim 1, comprising one of the at least one application agent for each class of application programs, each class of application programs being selected from the group consisting of electronic mail, fax, letter, file, operating system, address, games, flight simulators, vehicle simulators, naval simulators, sports simulators, war game and strategy simulators, role play simulators, action simulators, personal information manager, printer, calendar, terminal, travel, encyclopedia, image viewer, C++, Basic, graphical user interface, presentation, charting, meeting, scheduler, telephone, voice mail, word processor, spreadsheet, drawing, web, network, mathematical, directory assistance, internet retail sales, common household utility agent, K=12 education, general education, hands-on training for job-based tasks, internet event lookup, internet product-information lookup, internet-based meeting scheduler, and hardware manager.

11. A computer readable medium on which are stored instructions executable on a computer processor, the instructions comprising:

receiving natural language user input;

generating a parsed sentence from the natural language user input,

mapping the parsed sentence into a semantic action; and

generating an instruction from the semantic action, the instruction being executable by an application.

12. The computer readable medium of claim 11, wherein receiving natural language user input includes receiving natural language speech input.

13. The computer readable medium of claim 11, the instructions further comprising:

providing feedback to the user indicating the processor's understanding of the natural language user input; and

interacting with the user in natural language to clarify the natural language user input as necessary.

14. The computer readable medium of claim 13, wherein providing feedback to the user includes providing speech feedback to the user.

15. The computer readable medium of claim 11, the instructions further comprising generating a set of default conditions for executing the instruction by an application, the default conditions being specific to each user and/or common to multiple users.

16. The computer readable medium of claim 11, wherein the application is one of one or more applications selected from the group consisting of electronic mail, fax, letter, file, operating system, address, games, flight simulators, vehicle simulators, naval simulators, sports simulators, war game and strategy simulators, role play simulators, action simulators, personal information manager, printer, calendar, terminal, travel, encyclopedia, image viewer, C++, Basic, graphical user interface, presentation, charting, meeting, scheduler, telephone, voice mail, word processor, spreadsheet, drawing, web, network, mathematical, directory assistance, internet retail sales, common household utility agent, K-12 education, general education, hands-on-training for job-based tasks, internet event lookup, internet product-information lookup, internet-based meeting scheduler, and hardware manager.

17. The computer readable medium of claim 11, wherein the computer readable medium is selected from the group consisting of CD-ROM, zip disk, floppy disk, tape, flash memory, system memory, hard drive, and data signal embodied in a carrier wave.

18. A method for receiving, interpreting and executing natural language user input, comprising:

receiving natural language user input;

generating a parsed sentence from the natural language user input,

semantically interpreting the parsed sentence and generating an output command from the parsed sentence,

outputting the output command to an application class agent,

semantically interpreting the output command and generating an executable instruction from the output command, and

outputting the executable instruction to an application program for execution by the application program.

19. The method for receiving, interpreting and executing natural language user input of claim 18, wherein receiving natural language user input includes receiving natural language speech input.

20. The method for receiving, interpreting and executing natural language user input of claim 18, further comprising:

providing feedback to the user indicating the processor's understanding of the natural language user input; and

interacting with the user in natural language to clarify the natural language user input as necessary.

21. The method for receiving, interpreting and executing natural language user input of claim 20, wherein providing feedback to the user includes providing speech feedback to the user.

22. The method for receiving, interpreting and executing natural language user input of claim 18, further comprising generating a set of default conditions for executing the instruction by an application, the default conditions being specific to each user and/or common to multiple users.

23. The method for receiving, interpreting and executing natural language user input of claim 18, wherein semantically interpreting the parsed sentence and generating the output command includes mapping the parsed sentence into a semantic action as the output command.

24. The method for receiving, interpreting and executing natural language user input of claim 18, wherein semantically interpreting the output command and generating the executable instruction includes mapping the output command into a semantic action as the executable instruction.

25. The method for receiving, interpreting and executing natural language user input of claim 18, wherein the application is one of one or more applications selected from the group consisting of electronic mail, fax, letter, file, operating system, address, games, flight simulators, vehicle simulators, naval simulators, sports simulators, war game and strategy simulators, role play simulators, action simulators, personal information manager, printer, calendar, terminal, travel, encyclopedia, image viewer, C++, Basic, graphical user interface, presentation, charting, meeting, scheduler, telephone, voice mail, word processor, spreadsheet, drawing, web, network, mathematical, directory assistance, internet retail sales, common household utility agent, K-12 education, general education, hands-on training for job-based tasks, internet event lookup, internet product-information lookup, internet-based meeting scheduler, and hardware manager.

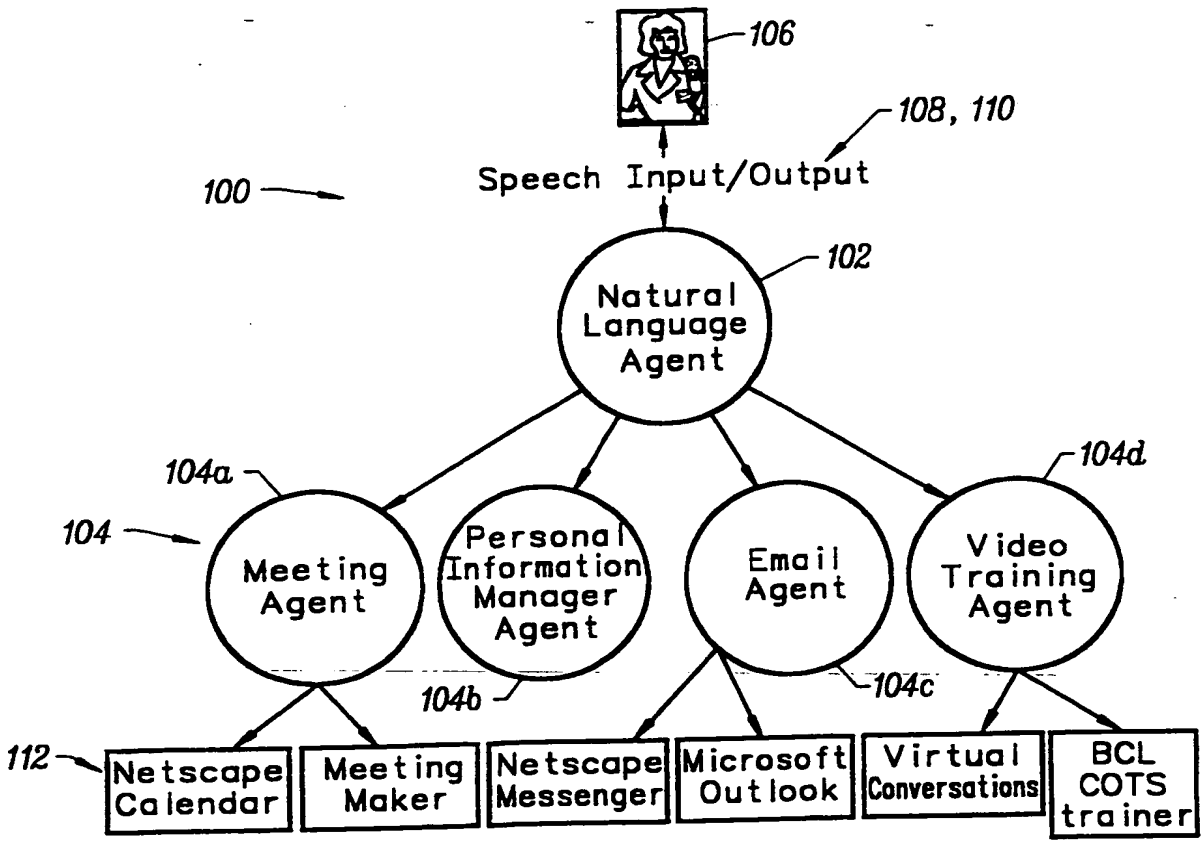


FIG. 1

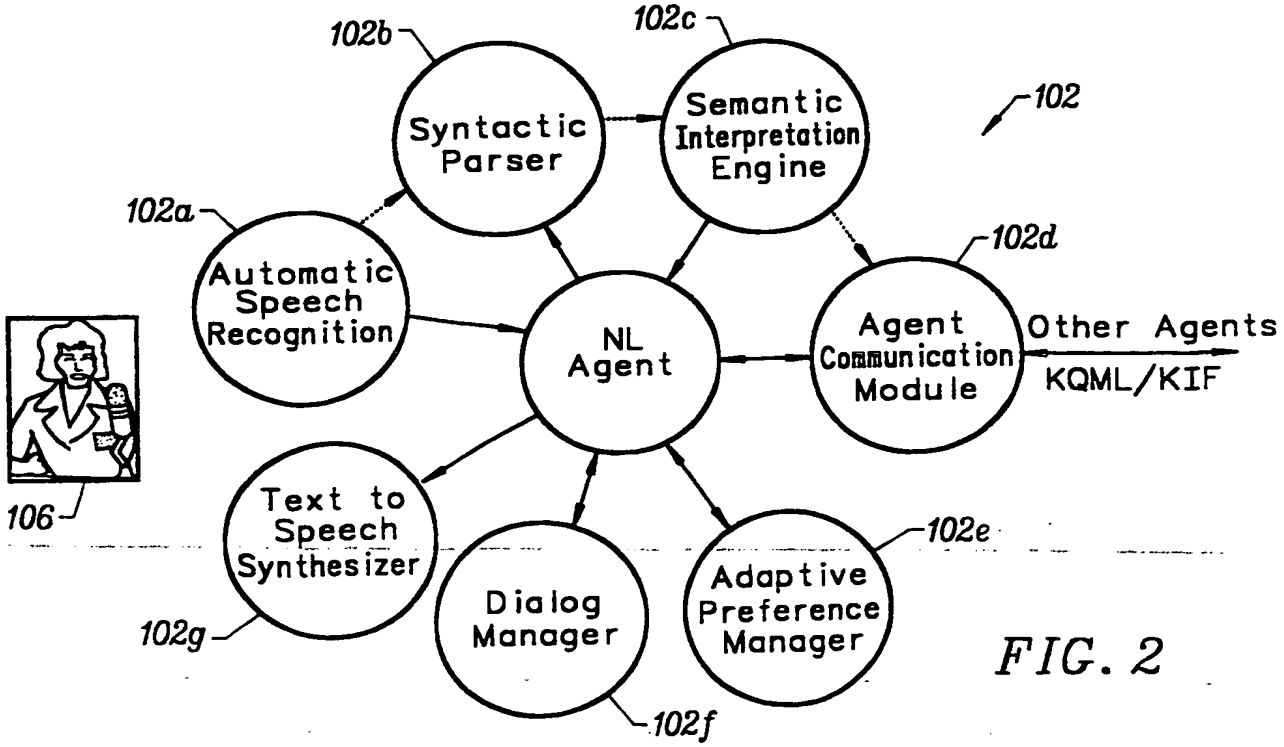


FIG. 2

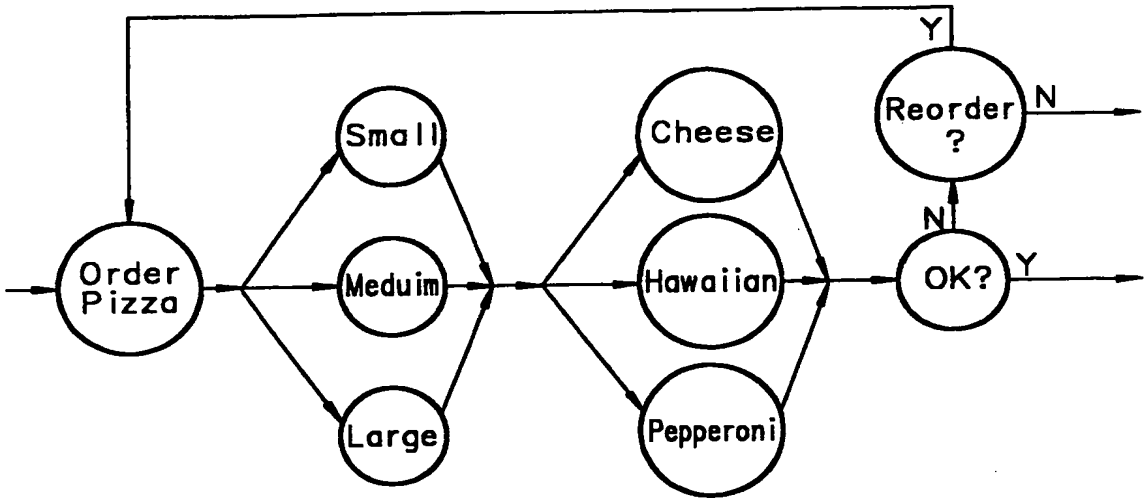


FIG. 3

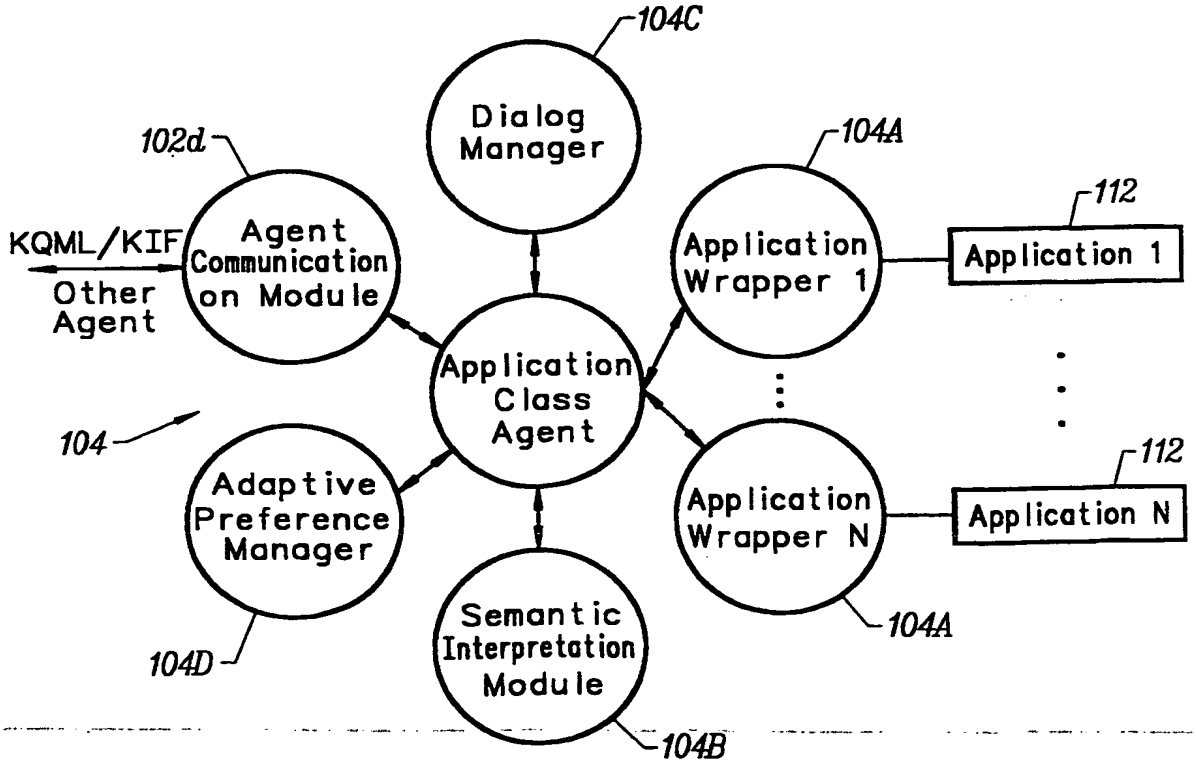


FIG. 4

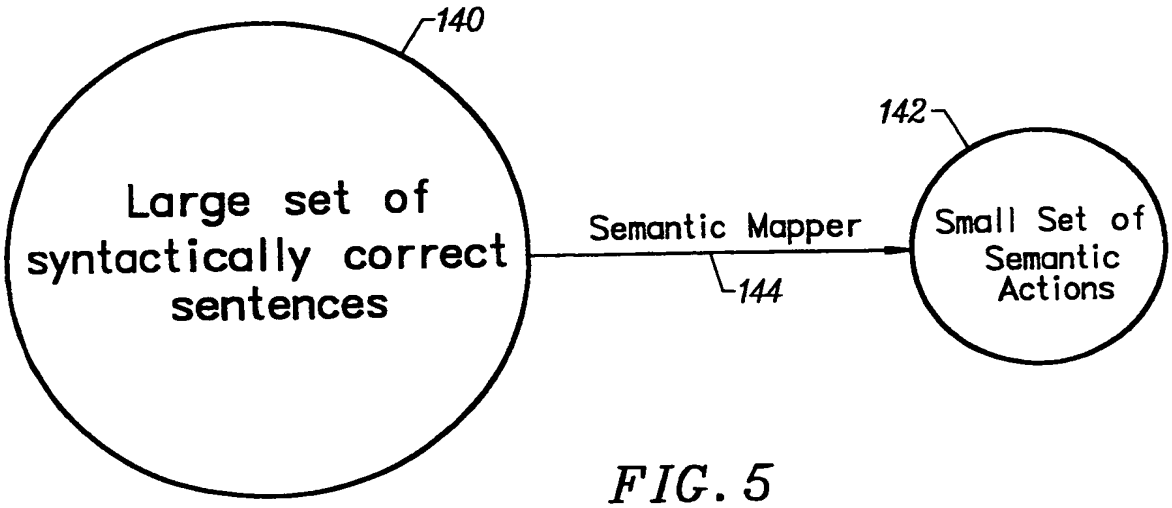


FIG. 5

ANALYTICAL Less Assertiveness and Less Responsivness	DRIVER More Assertiveness and Less Responsivness
AMIABLE Less Assertiveness and More Responsivness	EXPRESSIVE More Assertiveness and More Responsivness

FIG. 6

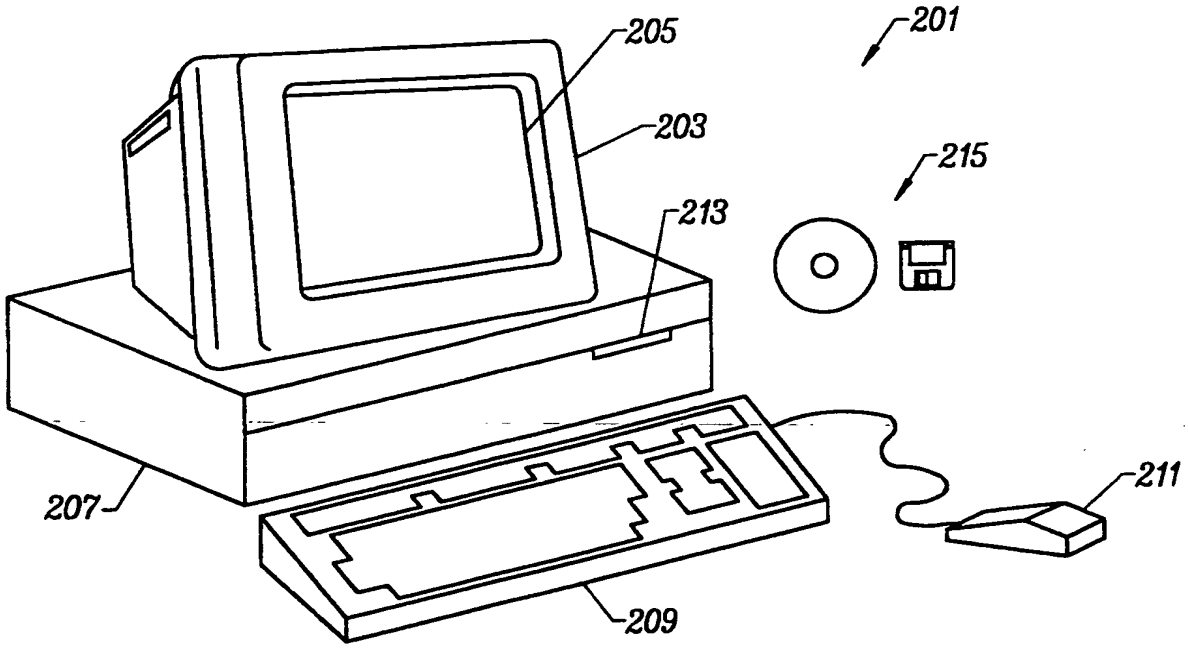


FIG. 7

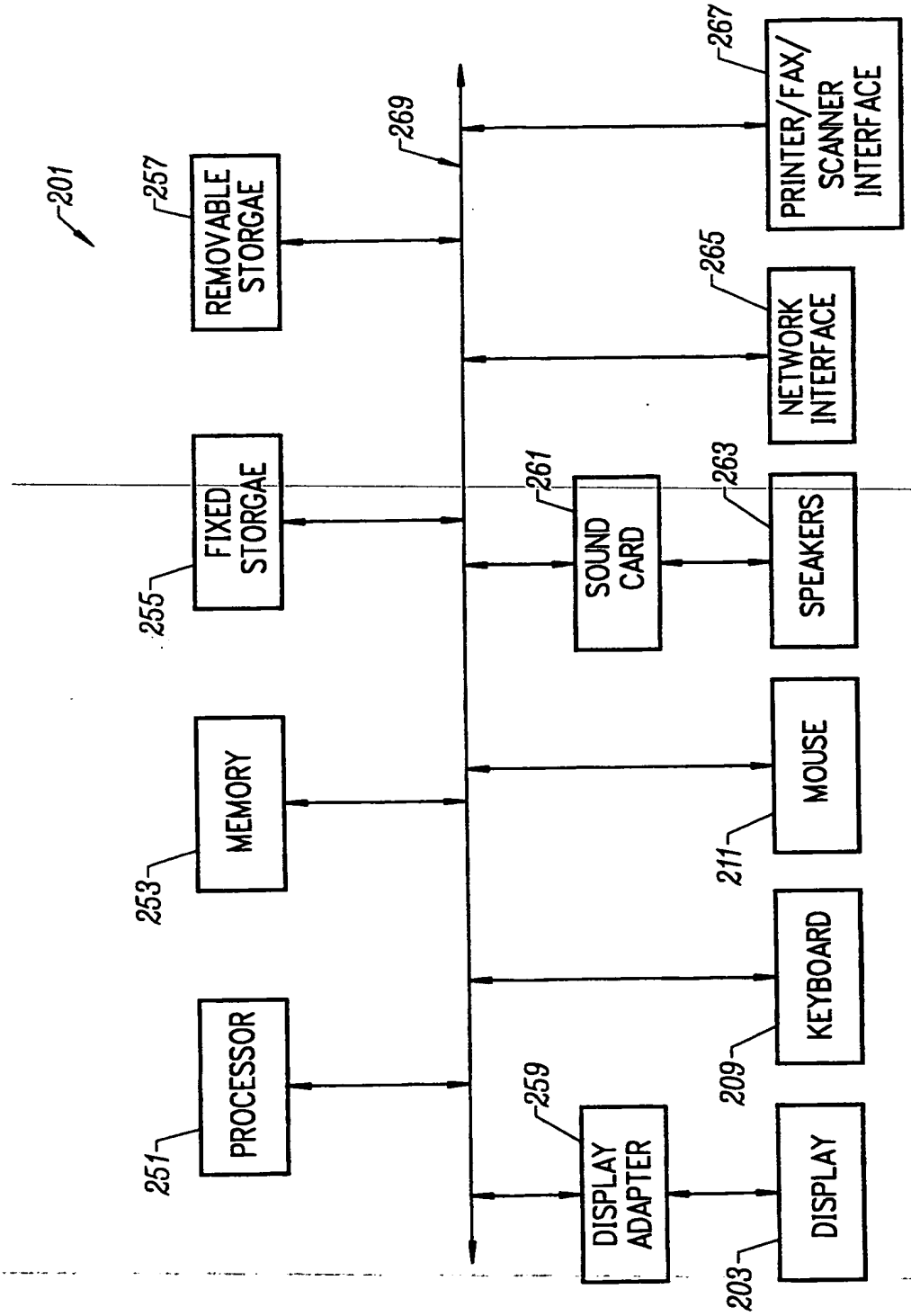


FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/19255

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 17/28, 17/27
US CL :704/9, 8, 257, 275

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 704/1, 9, 8, 10, 257, 275; 707/2, 3, 4, 5, 104, 530, 531, 532, 536

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X ----- A	US 5,748,974 A (JOHNSON) 05 May 1998, abstract, figs. 2-6; col. 1, line 60 to col. 2, line 32; and col. 2, line 61 to col. 6, line 40	11-25 ---------- 1-10
A	US 5,748,841 A (MORIN et al) 05 May 1998, abstract; figs. 1-8 & 11; col. 1, line 10 to col. 4, line 67; col. 5, line 34 to col. 20, line 64; and col. 22, lines 28-67	1-25
A	US 5,642,519 A (MARTIN et al) 24 June 1997 abstract, figs. 2-4; col. 4, line 39 to col. 5, line 33; col. 6, line 28 to col. 7, line 46; and col. 25, lines 35-67	1-25

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search
07 OCTOBER 1999

Date of mailing of the international search report
29 OCT 1999

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
JOSEPH THOMAS James R. Matthews
Telephone No. (703) 308-3900

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/19255

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,555,169 A (NAMBA et al) 10 September 1996, abstract; figs. 1-5, 9, 20, & 25; col. 1, line 9 to col. 3, line 8; col. 4, line 23 to col. 11, line 4; col. 12, line 51 to col. 15, line 20; and col. 26, line 6 to col. 29, line 63	1-25
A	US 4,688,195 A (THOMPSON et al) 18 August 1987, abstract, figs. 1-3; col. 1, line 10 to col. 7, line 68; col. 8, line 11 to col. 11, line 51; and col. 47, line 4 to col. 48, line 33	1-25